PURDUE UNIVERSITY | Information Technology RESEARCH COMPUTING

Biocontainers 101

Using Containers in Bioinformatics

ITaP Research Computing Virtual Workshop Series

Yucheng Zhang, Lev Gorenstein

Feb. 25, 2022

Yucheng Zhang
Senior Life Science Scientist
ITaP Research Computing

Outline

- What are containers and why should we use them?
- Singularity basics
- Public container repositories
- Pull and use public biocontainers
- Deployed biocontainers on RCAC clusters
- Build your own biocontainers

What are containers?

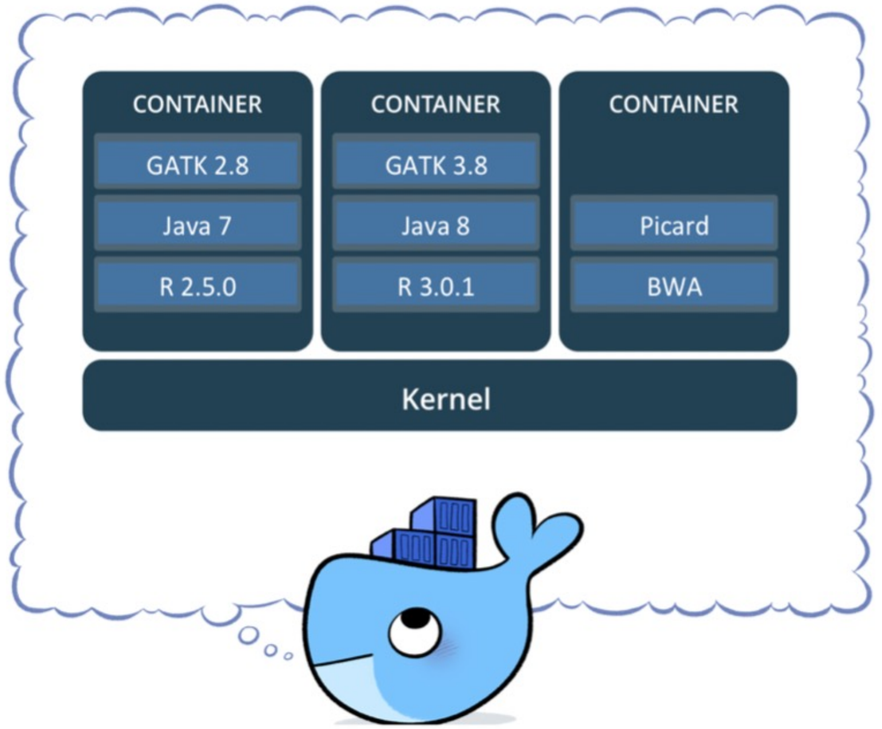
A **container** is an abstraction for a set of technologies that aim to solve the problem of how to get software to run reliably when moved from one computing environment to another.

A container **image** is simply a file (or collection of files) saved on disk that stores everything you need to run a target application or applications.

Registry is a place to store (and share) container images.

Why should we use containers?

- **Getting organized:** containers keep things organized by isolating programs and their dependencies inside containers.
- **Build once, run almost anywhere:** containers allow us to package up our complete software environment and ship it to numerous operating systems.
- **Reproducibility:** containers can ensure identical versions of apps, libraries, compliers, etc.



Source: https://support.terra.bio/hc/en-us/articles/360037340472-Docker-container-overview

A real example

```
whatis("Pipeline for profiling microbial pathways from sequencing data "
load("metaphlan2")
load("diamond")
load("MinPath")
load("RAPsearch")
load("usearch/8")
prepend_path("PATH","/group/bioinfo/apps/apps/humann2-0.11.1/bin")
prepend_path("PYTHONPATH","/group/bioinfo/apps/apps/humann2-0.11.1/lib/p
help([[
Notes:
HUMAnN2: The HMP Unified Metabolic Analysis Network 2
version 0.11.1
Pipeline for profiling microbial pathways from sequencing data
http://huttenhower.sph.harvard.edu/humann2
Changes: /group/bioinfo/apps/apps/humann2-0.11.1/history.md

Includes full databases from:
http://huttenhower.sph.harvard.edu/humann2_data/chocophlan/full_chocophlan_p
http://huttenhower.sph.harvard.edu/humann2_data/uniprot/uniref_annotated/uni

Forum: https://groups.google.com/forum/#!forum/humann-users
Manual: http://huttenhower.sph.harvard.edu/humann2/manual
Tutorial: https://bitbucket.org/biobakery/biobakery/wiki/humann2

]])
```

REQUIREMENTS

1. MetaPhlAn 2.0
2. Bowtie2 (version >= 2.1) (see NOTE)
3. Diamond (0.9.0 > version >= 0.8.22) (see NOTE)
4. MinPath (see NOTE)
5. Python (version >= 2.7)

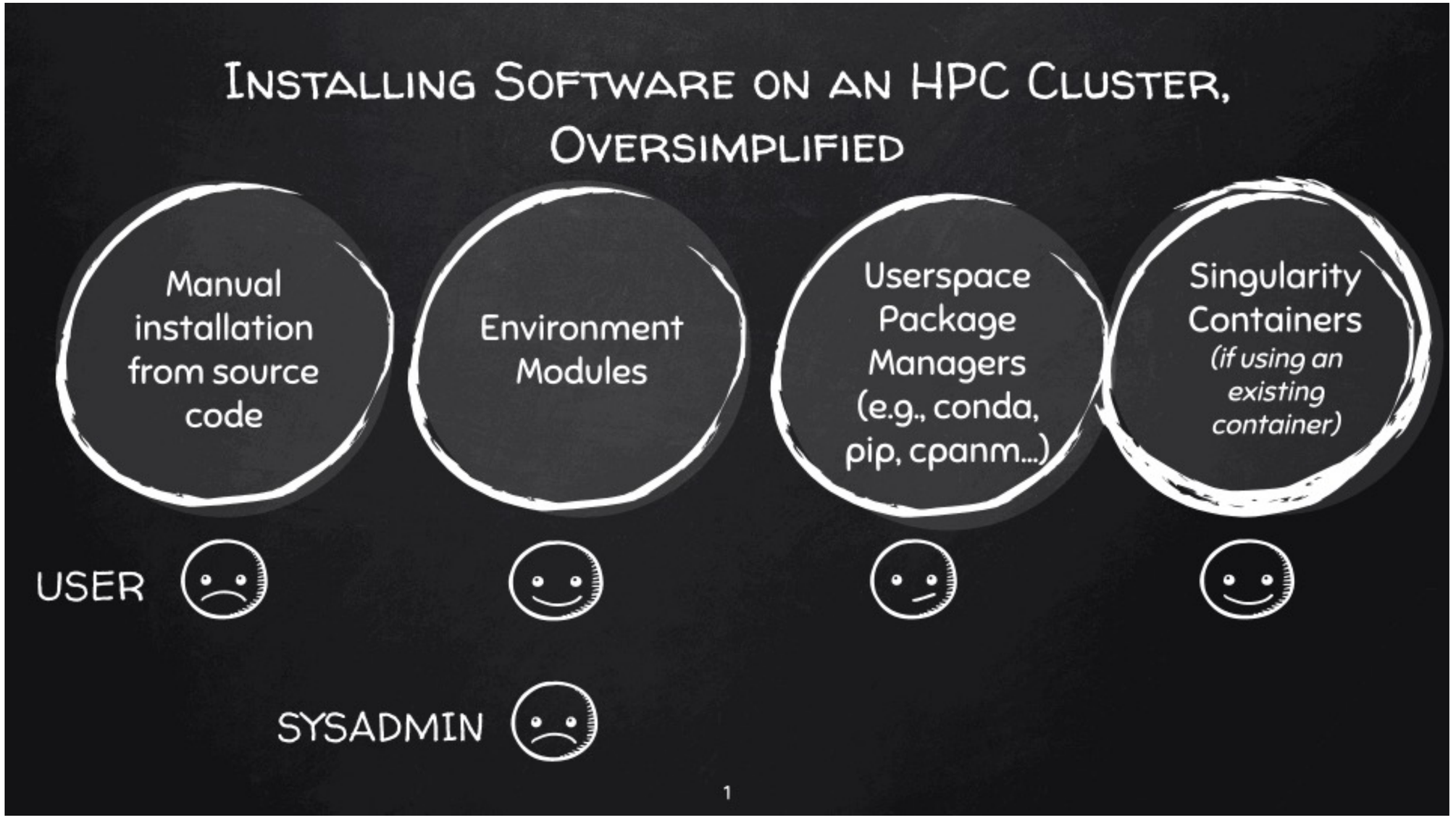
```
diamond/0.6.12
diamond/0.7.1
diamond/0.7.8
diamond/0.7.9
diamond/0.8.36
diamond/0.9.6
diamond/0.9.26                                    (D)
```

Why should we use containers on our clusters?

Enable you to install and use software easier

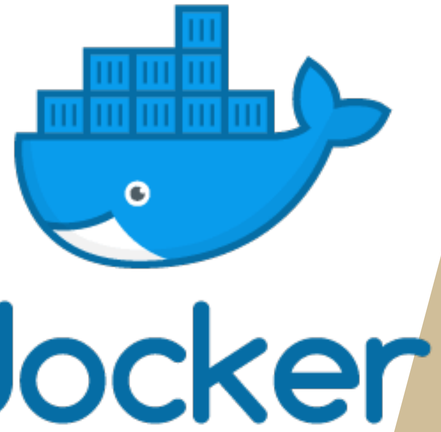
- Some software or packages has specific requirements for certain libraries such as **GLIBC**. Because our cluster's OS version is old, its libraries may not be compatible with your software. In such case, installing software into our clusters will be very challenging.

- However, if you build a container, you can get the latest everything and aren't limited by the cluster's OS version. **You are the master of your containers.**



source: https://github.com/harvardinformatics/bioinformatics-coffee-hour/blob/master/singularity/images/installing_software.gif

Docker



The concept of containers emerged in 1970s, but they were not well known until the emergence of Docker containers in 2013.

Docker is an open source platform for building, deploying, and managing containerized applications.

Some concerns about the security of Docker containers on HPC: Docker gives superuser privileges, but we do not want users to have full, unrestricted admin/ root access.

Singularity

- Singularity was developed in 2015 as an open-source project by researchers at Lawrence Berkeley National Laboratory led by Gregory Kurtzer.

- Singularity is emerging as the containerization framework of choice in HPC environments.
  1. Enable researchers to package entire scientific workflows, libraries, and even data.
  2. Users do not need to ask their system admin (e.g., RCAC) to install software for them.
  3. **Can use docker images.**
  4. Secure!
  5. **Does not require root privileges.**

Singularity basics

Detailed singularity user guide is available at: sylabs.io/guides/3.8/user-guide

The main singularity command

singularity [options] <subcommand> [subcommand options …]

- Build
- Pull
- Shell
- Exec

Singularity workflow on HPC

1 (Optional). **Build** singularity containers on a computer system where you have root or sudo privilege, e.g., your personal computer with singularity installed.

2. **Pull** the public containers or **transfer** your own containers to HPC.

3. **Run** singularity containers on the HPC system.

singularity pull

Download a container from a given URI.

singularity pull [output file] <URI>

Supported URIs include:

- **Library**: pull an image from singularity library
  library://<user>/<collection>/<image>[:tag]
- **Docker hub**: pull an image from Docker Hub.
  docker://<repository>/<image>[:tag]
- **Quay.io**: pull an image from Quay.io registry
  docker://quay.io/<repository>/<image>[:tag]
- **http, https**: pull an image using the http(s?) protocol
  e.g., https://library.sylabs.io/v1/imagefile/library/default/alpine:latest

Three useful image registries

1. **Docker Hub** (https://hub.docker.org)
   - Online repository of Docker container images.
   - As of Feb. 17, 2022, 8,842,825 available container images.
2. **BioContainers** (https://biocontainers.pro/registry)
   - A community-driven project for bioinformatics containers.
   - 10.6K tools,45.4K versions,222.5Kcontainers and packages.
   - The Bioconda package index lists all software available.
   - The Biocontainers registry provides a searchable interface.
3. **GALAXY project** (https://depot.galaxyproject.org/singularity/)
   - The BioContainers community also stored each singularity image in Galaxy depot.
   - Can be pulled or ran using the HTTP protocol.

singularity pull example

singularity pull [options] name_to_save.sif URI

Let's pull bowtie2 from three different resources.

- **Docker hub** (https://hub.docker.com/r/biocontainers/bowtie2/tags)

  singularity pull bowtie.2.4.1.sif docker://biocontainers/bowtie2:v2.4.1_cv1

- **Bioconda package index** (https://bioconda.github.io/recipes/bowtie2/README.html)

  singularity pull bowtie.2.4.5.sif docker://quay.io/biocontainers/bowtie2: 2.4.5--py37hafa4d4c_1

- **GALAXY project** (https://depot.galaxyproject.org/singularity)

  singularity pull bowtie.2.4.5.sif https://depot.galaxyproject.org/singularity/bowtie2:2.4.5--py39hbb4e92a_0

Recommendation: add the *--disable-cache* option to prevent image layers from being cached in ${HOME}/.singularity/cache

singularity pull --disable-cache name_to_save.sif URI

singularity shell

Go inside the container and start an interactive shell

singularity shell myimage.sif

```
zhan4429@brown-fe00:~ $ cat /etc/*release
CentOS Linux release 7.7.1908 (Core)
NAME="CentOS Linux"
VERSION="7 (Core)"
ID="centos"
ID_LIKE="rhel fedora"
VERSION_ID="7"
PRETTY_NAME="CentOS Linux 7 (Core)"
ANSI_COLOR="0;31"
CPE_NAME="cpe:/o:centos:centos:7"
HOME_URL="https://www.centos.org/"
BUG_REPORT_URL="https://bugs.centos.org/"
```

```
zhan4429@brown-fe00:~ $ singularity shell ubuntu_latest.sif
Singularity> cat /etc/*release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=20.04
DISTRIB_CODENAME=focal
DISTRIB_DESCRIPTION="Ubuntu 20.04.3 LTS"
NAME="Ubuntu"
VERSION="20.04.3 LTS (Focal Fossa)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 20.04.3 LTS"
VERSION_ID="20.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=focal
UBUNTU_CODENAME=focal
Singularity>
```

Type “**exit**” in the interactive shell to go back to host system

Bind mounts

- Programs running inside a container will not have access to directories and files outside of your home and the current directory.
- Singularity allows you to map directories on your host system to directories within your container using bind mounts.

```
singularity shell --bind hostdir1:containerdir1 --bind hostdir2:containerdir2 myimage.sif
```

Singularity binds several directories into the container image automatically. **$HOME**, **/tmp** and **$PWD** is the default list.

We also configured singularity to bind **/apps**, **/depot**, and **/scratch** on our clusters.

singularity exec

Run a command within a container

singularity exec myimage.sif **command**

For example:

singularity exec blast.2.11.0.sif blastx -query input.fasta -db swissprot -out blast.out

--bind option is also very useful for singularity exec

For example:

singularity exec --bind $HOME/data/:/data/ blast.2.11.0.sif blastx -query /data/input.fasta -db nr

input.fasta is located in the host directory $HOME/data/

RCAC Biocontainers

---------- BioContainers collection modules ----------

abyss/2.3.2		canu/2.2	(D)	gatk4/4.2.5.0	(D)	lra/1.3.2		pilon/1.24		snakemake/6.8.0	
abyss/2.3.4	(D)	cd-hit/4.8.1		genemark/4.68		macs2/2.2.7.1-py39		pirate/1.0.4		snap-aligner/2.0.0	
afplot/0.2.1-py36		cellranger-atac/2.0.0		genomicconsensus/2.3.3		macs3/3.0.0a6		plink/1.90b6.21		snap/2013_11_29	
alien-hunter/1.7.7		cellranger/6.0.1		genrich/0.6.1		mafft/7.475		plink2/2.00a2.3		snaptools/1.4.8	
allpathslg/52488		cellranger/6.1.1	(D)	glimmer/3.02		mafft/7.490	(D)	popscle/0.1b		soapdenovo2/2.40	
alphafold/2.1.1		circos/0.69.8		glimmerhmm/3.0.4		maker/2.31.11		prodigal/2.6.3		sortmerna/2.1b	
anchorwave/1.0.1		clustalw/2.1		gmap/2021.05.27		maker/3.01.03	(D)	prokka/1.14.6		sortmerna/4.3.4	(D)
angsd/0.935		cnvkit/0.9.9-py		gmap/2021.08.25	(D)	mauve/2.4.0		pyani/0.2.11		spaceranger/1.3.0	
annovar/2022-01-13		concoct/1.1.0-py38		goatools/1.1.12		mcl/14.137-p15262		qiime2/2021.2		spades/3.15.3	
antismash/5.1.2		coverm/0.6.1		graphlan/1.1.3		megahit/1.2.9		quast/5.0.2-py37		sra-tools/2.11.0-p15262	
antismash/6.0.1	(D)	csvtk/0.23.0		gtdbtk/1.7.0		meme/5.3.0-py37		quickmirseq/1.0		stacks/2.60	
atram/2.4.3		cufflinks/2.2.1-py36		hicexplorer/3.7.2		metabat/2.15-5		r-rnaseq/4.1.1-1-rstudio		star/2.7.9a	
augustus/3.4.0		cutadapt/3.4-py38		hifiasm/0.16.0		metaphlan/3.0.9		r-rnaseq/4.1.1-1	(D)	star/2.7.10a	(D)
bamtools/2.5.1		cyvcf2/0.30.14-py37		hisat2/2.2.1		metaphlan/3.0.14	(D)	r-rstudio/4.1.1		stringtie/2.1.7	
barrnap/0.9.4		dbg2olc/20180222		hmmer/3.3.2		minialign/0.5.3		r-scrnaseq/4.1.1-1-rstudio		stringtie/2.2.1	(D)
bbmap/38.93		deeptools/3.5.1-py		homer/4.11		miniasm/0.3_r179		r-scrnaseq/4.1.1-1	(D)	subread/1.6.4	
bcftools/1.13		diamond/2.0.13		htseq/0.13.5-py36		minimap2/2.22		r/4.1.1		subread/2.0.1	(D)
bcftools/1.14	(D)	diamond/2.0.14	(D)	htseq/0.13.5-py37		mirdeep2/2.0.1.3		racon/1.4.20		talon/5.0	
beagle/5.1_24Aug19.3e8		dsuite/0.4.r43		htseq/0.13.5-py38		mmseqs2/13.45111		raxml-ng/1.1.0		taxonkit/0.9.0	
beast2/2.6.3		edta/1.9.6		htseq/1.99.2-py37	(D)	mothur/1.46.0		raxml/8.2.12		tiara/1.0.2	
bedtools/2.30.0		edta/2.0.0	(D)	htslib/1.14		mothur/1.47.0	(D)	repeatmasker/4.1.2		tophat/2.1.1-py27	
bioawk/1.0		emboss/6.6.0		htstream/1.3.3		mrbayes/3.2.7		repeatmodeler/2.0.2		tpmcalculator/0.0.3	
biopython/1.70-np112py27		epic2/0.0.51-py39		humann/3.0.0		multiqc/1.11		repeatscout/1.0.6		tpmcalculator/0.0.4	(D)
biopython/1.70-np112py36		evidencemodeler/1.1.1		idba/1.1.3		mummer4/4.0.0rc1-p15262		rmats/4.1.1-py37		transdecoder/5.5.0	
biopython/1.78	(D)	exonerate/2.4.0		igv/2.11.9		muscle/3.8.1551		rmats2sashimiplot/2.0.4-py37		trim-galore/0.6.7	
bismark/0.23.0		fastani/1.32		impute2/2.3.2		nanofilt/2.8.0		rsem/1.3.3		trimal/1.4.1	
blasr/5.3.5		fastp/0.20.1		iqtree/1.6.12		nanolyse/1.2.0		rseqc/4.0.0-py37		trimmomatic/0.39	
blast/2.11.0		fastp/0.23.2	(D)	iqtree/2.1.2		nanoplot/1.39.0		run_dbcan/3.0.2		trinity/2.12.0	
blobtools/1.1.1		fastqc/0.11.9		iqtree/2.2.0_beta	(D)	nextflow/21.10.0		rush/0.4.2		trinity/2.13.2	(D)
bmge/1.12		faststructure/1.0-py27		isoseq3/3.4.0		orthofinder/2.5.2		salmon/1.5.2		trnascan-se/2.0.9	
bowtie/1.3.1-py38		fasttree/2.1.10		kallisto/0.46.2		orthofinder/2.5.4	(D)	salmon/1.6.0	(D)	varscan/2.4.2	
bowtie2/2.4.2-py38		fastx_toolkit/0.0.14		khmer/3.0.0a3-py36		paml/4.9		samtools/1.9		varscan/2.4.4	(D)
bracken/2.6.1-py37		filtlong/0.2.1		kmer-jellyfish/2.3.0		panphlan/3.1		scanpy/1.8.2		vcftools/0.1.16	
braker2/2.1.6		flye/2.9-py38		kneaddata/0.10.0		pbmm2/1.7.0		scvelo/0.2.4		velvet/1.2.10	
busco/5.2.2-py		fraggenescan/1.31		kraken2/2.1.2		perl-bioperl/1.7.2-p1526		seidr/0.14.2		vsearch/2.19.0	
bwa/0.7.17		freebayes/1.3.5-py38		lambda/2.0.0		picard/2.25.1		seqkit/2.0.0		weblogo/3.7.8	
cafe/4.2.1		fseq/2.0.3		lima/2.2.0		picard/2.26.10	(D)	seqkit/2.1.0	(D)		
canu/2.1.1		gatk4/4.2.0		lofreq/2.1.5		picrust2/2.4.2		skewer/0.2.2			

For example:

$ module load biocontainers

$ module load bamtools/2.5.1

$ bamtools -h

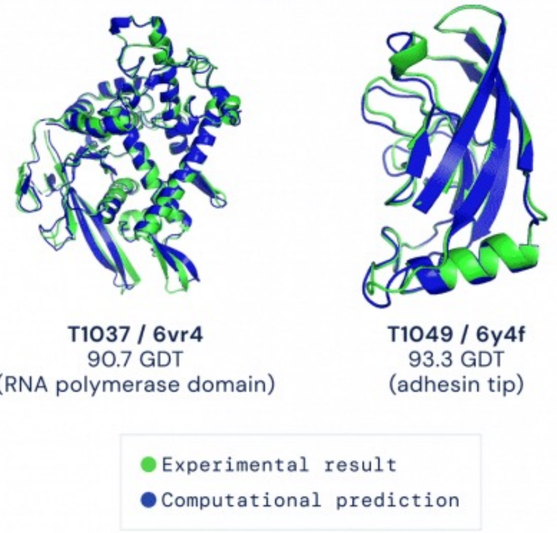
A collection of pre-downloaded container images wrapped into handy modulefiles so they look and feel like native applications.

Alphafold

Deployed in all clusters, support both CPU and GPU.

$ module load biocontainers

$ module load alphafold/2.1.1



The full database (~2.2TB) has been downloaded and setup for users.

Usage:

run_alphafold.sh --flagfile=$AlphaDB **--fasta_paths**=XX **--output_dir**=XX ...

$AlphaDB (/depot/itap/datasets/alphafold/full_db.ff) is a configuration file passed to AlphaFold containing the location of the database. Typically it should not be edited. Users can add other parameters based on your needs.

https://github.com/deepmind/alphafold

HOMER

Software for motif discovery and next-gen sequencing analysis

$ module load biocontainers
$ module load homer/4.11

Selected database have been downloaded for users.

ORGANISMS: yeast, worm, mouse, arabidopsis, zebrafish, rat, human and fly.

PROMOTERS: yeast, worm, mouse, arabidopsis, zebrafish, rat, human and fly.

GENOMES: hg19, hg38, mm10, ce11, dm6, rn6, danRer11, tair10, and sacCer3.

Check installed databases:

$ configureHomer.pl -list

PURDUE UNIVERSITY | Information Technology RESEARCH COMPUTING

GTDB TK 1.7.0

Toolkit for assigning objective taxonomic classifications to bacterial and archaeal genomes based on the Genome Database Taxonomy GTDB.

```
$ module load biocontainers
$ module load gtdbtk/1.7.0
```

GTDB-Tk reference data (R202) has been downloaded for users.

Example usage:

```
$ gtdbtk identify --genome_dir genomes --out_dir identify --extension gz --cpus 8

$ gtdbtk align --identify_dir identify --out_dir align --cpus 8

$ gtdbtk classify --genome_dir genomes --align_dir align --out_dir classify --extension gz --cpus 8
```

HUMAnN 3

Quantify species' contributions to community function

$ module load biocontainers

$ module load humann/3.0.0

Full **ChocoPhlAn**, **UniRef90**, **EC-filtered UniRef90**, **UniRef50**, **EC-filtered UniRef50**, and **utility_mapping** databases have been downloaded for users.

Check the database and config by:

$ humann_config --print

HUMAnN Configuration (Section : Name = Value)
database_folders : nucleotide = /depot/itap/datasets/humann/chocophlan
database_folders : protein = /depot/itap/datasets/humann/uniref
database_folders : utility_mapping = /depot/itap/datasets/humann/utility_mapping

Run_dbcan

Automated CAZyme annotation

$ module load biocontainers

$ module load run_dbcan/3.0.2

Latest version of database has been downloaded and setup, including **CAZyDB.09242021.fa**, **dbCAN-HMMdb-V10.txt, tcdb.fa, tf-1.hmm, tf-2.hmm**, and **stp.hmm**.

Usage:

$ **run_dbcan** protein.faa protein **--out_dir** test1_dbcan

$ **run_dbcan** genome.fasta prok **--out_dir** test2_dbcan

https://github.com/linnabrown/run_dbcan

R-RNAseq

Customized R container for RNAseq analysis.

```
$ module load biocontainers
$ module load r-rnaseq/4.1.1-1
OR $ module load r-rnaseq/4.1.1-1-rstudio
```

Commands:

1. R
2. Rscript
3. rstudio (only exist in rstudio version)

BiocManager 1.30.16	readr 2.0.2
ComplexHeatmap 2.9.4	readxl 1.3.1
DESeq2 1.34.0	purrr 0.3.4
edgeR 3.36.0	dplyr 1.0.7
DEXSeq 1.40.0	stringr 1.4.0
pheatmap 1.0.12	
limma 3.48.3	forcats 0.5.1
tibble 3.1.5	ggplot2 3.3.5
tidyr 1.1.4	openxlsx 4.2.5

Thanks to Lev Gorenstein's r/4.1.1 base image, users can also install other packages, same with non-containerized R.

PURDUE UNIVERSITY | Information Technology RESEARCH COMPUTING

R-scRNAseq

Customized R container for scRNAseq analysis.

$module load biocontainers

$module load r-scrnaseq/4.1.1-1

OR $module load r-scrnaseq/4.1.1-1-rstudio

Commands:

1. R
2. Rscript
3. rstudio (only exist in rstudio version)

BiocManager 1.30.16
Seurat 4.1.0
SeuratObject 4.0.4
SeuratWrappers 0.3.0
monocle3 1.0.0
SingleCellExperiment 1.16.0
scDblFinder 1.8.0
SingleR 1.8.1
scCATCH 3.0
scMappR 1.0.7
rliger 1.0.0

schex 1.8.0
CoGAPS 3.14.0
celldex 1.4.0
dittoSeq 1.6.0
DropletUtils 1.14.2
miQC 1.2.0
Nebulosa 1.4.0
tricycle 1.2.0
pheatmap 1.0.12
limma 3.48.3, 3.50.0
tibble 3.1.5

tidyr 1.1.4
readr 2.0.2
readxl 1.3.1
purrr 0.3.4
dplyr 1.0.7
stringr 1.4.0
forcats 0.5.1
ggplot2 3.3.5
openxlsx 4.2.5

Thanks to Lev Gorenstein's r/4.1.1 base image, users can also install other packages, same with non-containerized R.

Build your own containers with singularity

The first step is to install singularity on your personal computer.

We have singularity version **3.8.0** on the cluster. To guarantee compatibility, please be sure to follow the installation guide for version 3.8 on your system (https://sylabs.io/guides/3.8/user-guide/quick_start.html).

$ **sudo singularity build** image.sif **image.def**

- Need to build using a computer with elevated privileges, then copy to cluster.
- If no access to such a computer, can also build in the cloud.

Remote builder

If you need to build an image from a system where you don't have admin privileges, we can build remotely using the Sylabs Remote Builder.

To remotely build an image using singularity, go through the following steps:

1. Go to: https://cloud.sylabs.io/, and create a Sylabs account.
2. Create a new "Access Token", and copy it to clipboard.
3. Login to our clusters, and run `singularity remote login` in terminal and paste the access token at the prompt.
4. Then you can remotely build your own singularity image on the cluster.

```
singularity build -r myimage.sif myimage.def
```

or

```
singularity build --remote myimage.sif myimage.def
```

Once finished, the image will be downloaded automatically so that it's ready to use.

Singularity definition file

A **definition** file, or **def** file, is a recipe to build a container image with singularity. It is divided into two parts:

1. **Header**: the Header describes the core operating system to build within the container.
2. **Section**: each section is defined by a % character followed by the name of the particular section. Different sections add different content or execute commands at different times during the build process.

Detailed instruction on how to prepare a definition file is available at https://sylabs.io/guides/latest/user-guide/definition_files.html.

```
BootStrap: docker
From: debian:buster-slim
```

Header

```
##################################################################
%post
##################################################################
apt-get -y update
apt-get install -y curl wget nano bzip2 less

# miniconda
mkdir -p /opt
wget https://repo.continuum.io/miniconda/Miniconda3-latest-Linux-x86_64.sh
bash Miniconda3-latest-Linux-x86_64.sh -f -b -p /opt/conda
. /opt/conda/etc/profile.d/conda.sh
conda activate base
conda update --yes --all
conda config --add channels bioconda
conda config --add channels conda-forge
conda create -n prokka prokka==1.14.6


# create bind points for NIH HPC environment
mkdir /gpfs /spin1 /data /scratch /fdb /lscratch /vf
for i in $(seq 1 20); do ln -s /gpfs/gsfs$i /gs$i; done

# clean up
apt-get clean
conda clean --yes --all

##################################################################
%environment
##################################################################
export LC_ALL=C
export PATH=/opt/conda/envs/prokka/bin:$PATH
```

Section

def file for prokka 1.14.6 prepared by NIH HPC staff

Preferred bootstrap agents

1. **library**: images hosted in Sylabs Cloud Library
2. **docker**: images hosted in Docker Hub
3. **localimage**: images saved on your machine

Information about more bootstrap agents can be found in Singularity user guide.

ANGSD

Program for analyzing NGS data

```
Bootstrap: docker
From: ubuntu:20.04
%labels
Author "Yucheng Zhang <zhan4429@purdue.edu>"
Version v0.935
%help
Singularity container with ANGSD v0.935. This container also installed htslib and samtools.

%post
# update the system and install building essentials
apt-get -y update
apt-get -y install --no-install-recommends --no-install-suggests libssl-dev libcurl4-gnutls-dev libbz2-dev  \
        liblzma-dev libz-dev samtools gcc g++ git ca-certificates build-essential  make zip wget unzip locales locales-all
# clean up
apt-get -y autoremove && apt-get clean. && rm -rf /var/lib/apt/lists/*
# Install htslib
SRC=/usr/local/src
mkdir -p $SRC && cd $SRC
git clone --recursive https://github.com/samtools/htslib.git
cd htslib && make
# Install angsd
cd $SRC && git clone https://github.com/ANGSD/angsd.git
cd angsd && make HTSSRC=$SRC/htslib
#Symbolic link
chmod +x $SRC/angsd/misc/realSFS
cd /usr/local/bin
ln -s $SRC/angsd/angsd .  && ln -s $SRC/angsd/misc/realSFS .
```

aTRAM

automated target restricted assembly method

```
Bootstrap: docker
From: continuumio/miniconda3

%labels
Author "Yucheng Zhang <zhan4429@purdue.edu>"
Version 2.4.3

%help
This container contains the latest version (v2.4.3) of aTRAM.

%post
conda install git
cd /opt/   &&  git clone https://github.com/juliema/aTRAM.git
cd aTRAM   &&   chmod +x *.py

conda install python=3 numpy biopython psutil
conda install -c bioconda  blast velvet trinity abyss spades exonerate

%environment
export PATH=/opt/aTRAM/:$PATH
```

```
Bootstrap: localimage
From: r-base:4.1.1.sif

%post
Rscript -e "install.packages('tidyverse')"

Rscript -e "install.packages('openxlsx', dependencies = TRUE)"

Rscript -e 'if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")' \
&& Rscript -e
'BiocManager::install(c("limma","edgeR","DESeq2","ComplexHeatmap","DEXSeq"))'
```

If users want to build your own R containers, welcome to use our r-base images and recipes that are stored in /depot/itap/biocontainers/recipes/


```
Bootstrap: localimage
From: r_4.1.1_rstudio.sif

%post
# update the system and install building essentials
apt-get -y update
apt-get install -y gdal-bin libgdal-dev libudunits2-dev  ## here you install required libraries
# clean up
apt-get -y autoremove && apt-get clean && rm -rf /var/lib/apt/lists/*

## Seurat3
Rscript -e "install.packages('Seurat')"

## monocle3
Rscript -e 'if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")' \
&& Rscript -e "BiocManager::install(c('BiocGenerics', 'DelayedArray', 'DelayedMatrixStats','limma',
'S4Vectors', 'SingleCellExperiment','SummarizedExperiment', 'batchelor','Matrix.utils'))"
Rscript -e "devtools::install_github('cole-trapnell-lab/leidenbase')"
Rscript -e "devtools::install_github('cole-trapnell-lab/monocle3')"
```

Cluster user guide

Singularity section contains instructions for using Singularity on RCAC clusters.

Biocontainer collection section contains instructions and examples for running bioinformatic containers.

Email

rcac-help@purdue.edu is our email support address. Send us an email any time.

Coffee hour consultations

In response to COVID-19, we are temporarily switching all our Coffee Hour Consultations to online only (https://www.rcac.purdue.edu/coffee). We offer several slots (2:00 to 3:30pm) each afternoon (Monday to Thursday) for private one-on-one consultations or questions of up to 30 minutes.

Additional bioinformatic tools

Contact me (zhan4429@purdue.edu) or Lev (lev@purdue.edu), if you want additional software added into RCAC biocontainers.

Thank you!

Questions?